

Phát triển Ứng dụng WEB

MVC và Framework

Giảng Viên: ThS. Mai Xuân Hùng



Nội dung

- Tổ chức code bằng mô hình MVC
- Framework là gì
- Giới thiệu các Framework trong PHP
- Framework CodeIgniter
 - Giới thiệu CodeIgniter
 - Cách cài đặt CodeIgniter
 - Cấu hình CodeIgniter
 - Cách tổ chức code trong CodeIgniter
 - Các thư viện CodeIgniter
 - CSDL trong CodeIgniter
 - Ajax trong CodeIgniter
 - Giỏ hàng trong CodeIgniter



TỔ CHỨC CODE BẰNG MÔ HÌNH MVC



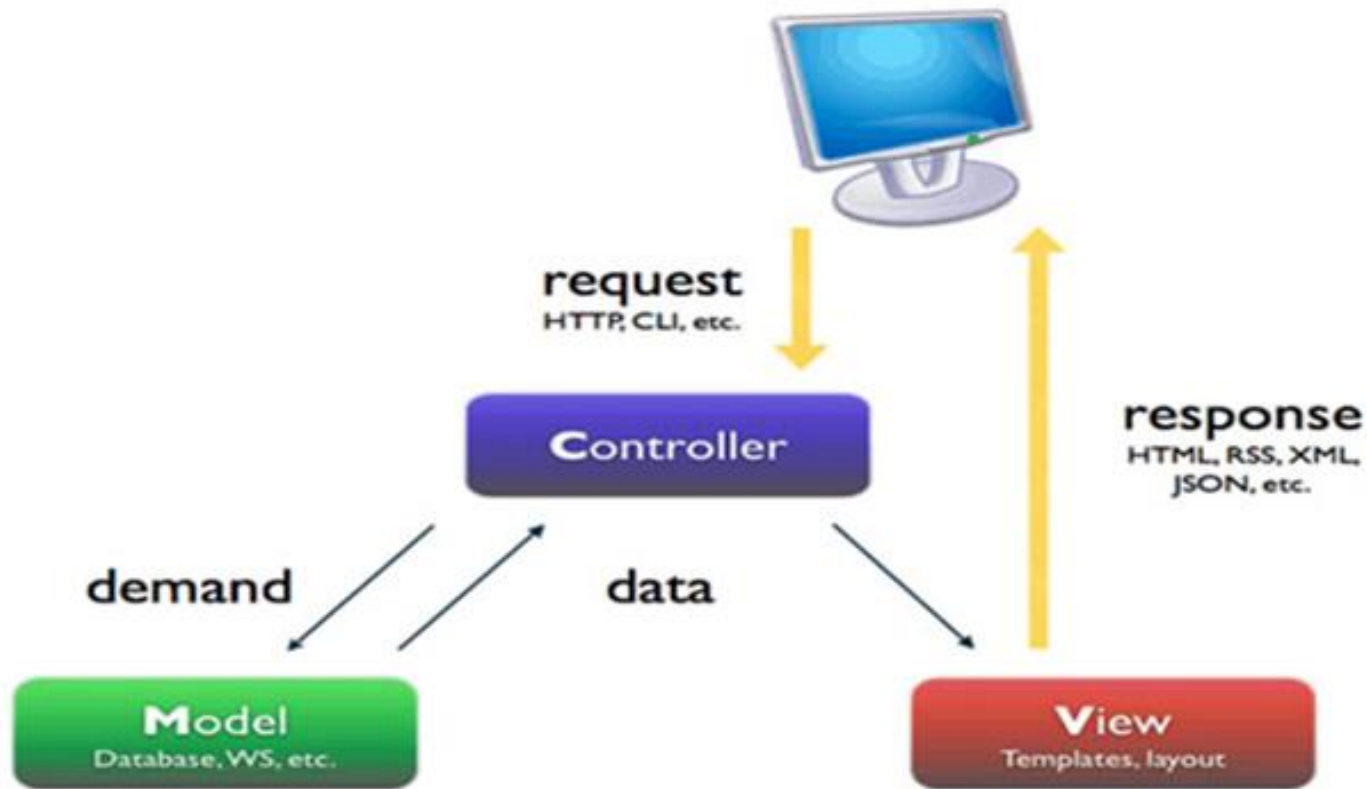
Giới thiệu



- MVC (Model – view - controller) là một mô hình tổ chức chương trình, cho phép lập trình viên tách ứng dụng thành 3 thành phần khác nhau Model, View và Controller.
- Mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với các thành phần khác
- MVC chia nhỏ quá trình xử lý của một ứng dụng, giúp người lập trình làm việc trên từng thành phần riêng lẻ, không ảnh hưởng đến các thành phần khác giúp phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp



Mô hình MVC



Model



- Model là thành phần chịu trách nhiệm tương tác với CSDL
- Tất cả các nghiệp vụ logic được thực thi ở Model.
- Dữ liệu vào từ người dùng sẽ thông qua View đến Controller và được kiểm tra ở Model trước khi lưu vào cơ sở dữ liệu.
- Việc **truy xuất, xác nhận, và lưu dữ liệu** là một phần của Model



View



- View làm nhiệm vụ hiển thị thông tin cho người dùng
- Nhận các dữ liệu vào từ người dùng, gửi đi các yêu cầu đến controller, sau đó là nhận lại các phản hồi từ controller và hiển thị kết quả cho người dùng.
- Trong các web framework, View gồm 2 phần chính:
 - Template file: Định nghĩa cấu trúc và cách thức trình bày dữ liệu cho user: ví dụ như layout, color, windows ...
 - Xử lý Logic: cách áp dụng dữ liệu vào cấu trúc trình bày. Logic này có thể bao gồm việc kiểm tra định dạng dữ liệu, chuyển đổi định dạng dữ liệu sang một dạng dữ liệu trung gian, lựa chọn một cấu trúc hiển thị phù hợp



Controller



- Controller đảm nhiệm việc nhận yêu từ người dùng thông qua View
- Xác định yêu cầu và gọi yêu cầu đến cho model xử lý
- Model gửi dữ liệu trả về sau khi xử lý xong cho Controller
- Controller nhận dữ liệu trả về từ model, chỉ định view hiển thị dữ liệu cho người dùng



Ưu điểm của mô hình MVC



- Thể hiện tính chuyên nghiệp trong lập trình, phân tích thiết kế.
- Do được chia thành các thành phần độc lập nên giúp phát triển ứng dụng:
 - Nhanh
 - Kết cấu phần mềm rõ ràng
 - Đơn giản
 - Dễ nâng cấp, bảo trì..



Nhược điểm của mô hình MVC

- Đối với dự án nhỏ việc áp dụng mô hình MVC gây
 - Công kênh
 - Tốn thời gian trong quá trình phát triển
 - Tốn thời gian trung chuyển dữ liệu của các thành phần

Ví dụ tổ chức code bằng mô hình MVC

Demo



Framework





Framework là gì

- Framework là phần mềm mã nguồn mở viết bằng ngôn ngữ lập trình web động
- Framework giúp các nhà phát triển phần mềm trong vấn đề tổ chức code khi triển khai các ứng dụng Web
- Đơn giản hóa khi tổ chức code
- Tổ chức code một cách logic



Framework là gì

- Framework dùng mô hình MVC để tổ chức code
- Dễ quản lý ứng dụng
- Giúp dễ sửa chữa, nâng cấp ứng dụng
- Giảm thời gian triển khai ứng dụng Web

Khi nào sử dụng Framework



- Với những lập trình viên mới bắt đầu, **framework** cung cấp những tính năng đơn giản và ổn định
- Framework sẽ giúp chúng ta giảm bớt, hoặc loại bỏ các **đoạn mã** thiếu tính khoa học, và **tăng tốc** cho quá trình xây dựng ứng dụng



Khi nào sử dụng Framework



- Đối với các lập trình viên PHP đã có kinh nghiệm, **framework** được xem như 1 công cụ giúp đỡ các lập trình viên viết code 1 cách **gọn gàng**, tốt hơn và khoa học hơn
- PHP framework là 1 công cụ được sử dụng để tiết kiệm thời gian và giúp cho việc tổ chức code chặt chẽ hơn, logic hơn



Những điểm lưu ý khi chọn 1 Framework



- Dễ sử dụng, phát triển nhanh và hiệu quả, phổ biến, có các tính năng mạnh mẽ, có diễn đàn hỗ trợ khi thảo luận.
- Hầu hết các **framework** đều có các điểm yếu và thế mạnh khác nhau:
- **Ví dụ:**
 - **CodeIgniter**: Gọn nhẹ, dễ sử dụng, được dùng nhiều
 - **Cake**: Ít người sử dụng hơn Zend nhưng thân thiện với người dùng và dễ sử dụng
 - **Zend**: Có đầy đủ các tính năng mạnh mẽ, cộng đồng người sử dụng lớn,



Một số Framework thông dụng

1. CodeIgniter
2. Laravel
3. Phalcon
4. Symfony
5. Yii 2
6. Cake PHP
7. Zend
8. Fat-Free
9. PHPixie
10. FuelPHP

Zend



The screenshot shows the Zend Framework 1.8 homepage. At the top, there is a navigation bar with links for 'ZF FRAMEWORK', 'ABOUT', 'DOWNLOADS', 'DOCUMENTATION', 'COMMUNITY', and 'SERVICES'. A search bar is located on the right. The main header features a large 'ZF 1.8' logo with a 'Download Now' button. Below the header, the page is divided into three columns: 'Make the Choice', 'Get Started', and 'Give Back'. The 'Make the Choice' column includes links to 'Zend Framework 1.5.0 Preview Release Now Available' and 'Zend Framework 1.8.101 Now Available', along with sections for 'Why ZF?', 'Case Studies', and 'Partners'. The 'Get Started' column includes links to 'Jerden Keppens on: Creating a modular application with Zend Framework', 'Benjamin Eberlei's Blog: Using a Dependency Injection Container with Zend_Application', 'QuickStart', 'Videos & Podcasts', and 'Reference Guide'. The 'Give Back' column includes links to 'PHP Performance Tips from Stas Malyshev', 'Preview of Windows Azure Support in Zend Framework Released', 'Contributors Guide', 'Issue Tracker', and 'Community Wiki'.

ZF 1.8

Introducing the Best Stack for your Zend Framework Applications

[Learn More](#)

Download Now

Make the Choice

Standardize your PHP practices

- [Zend Framework 1.5.0 Preview Release Now Available](#)
- [Zend Framework 1.8.101 Now Available](#)

Why ZF?
You've got questions, we've got solutions

Case Studies
From Fortune 500s to startups

Partners
See who's already on board

Get Started

What you need to get up to speed

- [Jerden Keppens on: Creating a modular application with Zend Framework](#)
- [Benjamin Eberlei's Blog: Using a Dependency Injection Container with Zend_Application](#)

QuickStart
Take our 30-minute tour

Videos & Podcasts
Your topics on your time

Reference Guide
More than 500 examples in 6 languages

Give Back

Because it just feels good

- [PHP Performance Tips from Stas Malyshev](#)
- [Preview of Windows Azure Support in Zend Framework Released](#)

Contributors Guide
Contributing is easier than you think

Issue Tracker
Help us deal with our issues

Community Wiki
The ultimate communal knowledge base



Zend



- Được xem Framework phổ biến nhất hiện nay
- Cộng đồng phát triển rộng lớn.
- Tập trung vào các ứng dụng web theo phong cách 2.0. Vì được phổ biến rộng rãi, và có 1 cộng đồng người dùng tích cực
- Tính năng mạnh mẽ, thường được sử dụng cho các công ty lớn
- Tập trung vào các ứng dụng web theo phong cách 2.0. Vì được phổ biến rộng rãi, và có 1 cộng đồng người dùng tích cực
- Cần phải có lượng kiến thức khá sâu rộng về PHP để có thể sử dụng được



CakePHP



The screenshot shows the CakePHP website homepage. At the top, there is a navigation bar with links: CakePHP, API, Docs, Bakery, Live, Forge, Trac, About CakePHP, and Developers. Below this is a secondary navigation bar with links: Jobs, Planet, Downloads, and Screenscasts, along with a Google Custom Search box. The main header features the CakePHP logo (a 3D cake with a slice missing) and the text: "CakePHP enables PHP users at all levels to rapidly develop robust web applications." Below the header is a red banner announcing "Extra Hot: Release: 1.2.3.8166 Stable". A dark blue navigation bar contains links: Get it now!, Hot Features, Learn, Interact, and Read. The main content area is divided into two columns. The left column features a large 3D box with the number "1.2" and the text "Get it now!". The right column contains a "Download 1.2.3.8166 Stable" button and a list of links: read the announcement, view the changelog, and discover the hot features. At the bottom right of the right column is a "I USE IT" button.



CakePHP



- Sự lựa chọn tuyệt vời cho những lập trình viên có kiến thức nâng cao về PHP
- Dựa trên cùng 1 nguyên tắc thiết kế với Ruby on Rails
- Giúp lập trình viên đẩy nhanh quá trình phát triển ứng dụng
- Với các hệ thống hỗ trợ, tính đơn giản và mỗi trường mở cao đã giúp cho CakePHP trở thành 1 trong những **framework** phổ biến nhất hiện nay.



Symfony



Symfony Web PHP framework

About Installation Documentation Plugins Community Blog Development

Open-Source PHP Web Framework

Symfony is a full-stack framework, a library of cohesive classes written in PHP.

It provides an architecture, components and tools for developers to build complex web applications faster. Choosing symfony allows you to release your applications earlier, host and scale them without problem, and maintain them over time with no surprise.

Symfony is based on experience. It does not reinvent the wheel: it uses most of the best practices of web development and integrates some great third-party libraries.

Thousands of developers already trust symfony for their applications!

Now users join the community every day, and that makes of symfony the most popular PHP framework around. A large community means easy-to-find support, user-contributed documentation, plugins, and free applications.

Getting Started

- Discover symfony with the [dedicated tutorial](#).
- Learn the framework with the [practical symfony book](#).
- Discuss on our [mailing lists](#), and join us in the [#symfony](#) IRC channel.

VOTE FOR symfony

COMMUNITY CHOICE AWARD 2011

symfony Releases

- 1.0 branch: 1.0.20
- 1.2 branch: 1.2.0
- 1.3 branch: dev

symfony Books

Learn symfony by the example

Search

powered by google

easy ajax

Watch the [ten minutes screencast](#) of the [ajax tutorial](#) or [test it online](#)

admin generator

This [screencast](#) shows the best PHP code generation engine for your backend interfaces.

Jobeet

24 days with...

The [Jobeet website](#) is a real-world web app, developed entirely in symfony. Step by step, [this is how](#).

book

In addition to [tutorials and showcase apps](#), symfony has an extensive documentation. [Click on...](#)



Symfony




- Giúp nâng cao hơn cho những lập trình viên muốn tạo ra các ứng dụng website doanh nghiệp
- Đây là 1 PHP **framework** mã nguồn mở với đầy đủ các tính năng cần thiết
- Nhưng nó có vẻ chạy chậm hơn các framework khác



Seagull



Search:

 **Seagull**

[Give it a go and try the Demo](#)

[Download Now](#)
STABLE 0.6.6 [2009-01-26]
for Windows, Linux or Mac

Syndicate

-
-
-
-
-

[Overview](#) [Download](#) [Docs](#) [Developers](#) [Forum](#) [Community](#)

Overview

Introduction

Seagull is a mature OOP framework for building web, command line and GUI applications. Licensed under BSD, the project allows PHP developers to easily integrate and manage code resources, and build complex applications quickly.

Many popular PHP applications are already seamlessly integrated within the project, as are various templating engines, testing tools and managed library



Seagull



Phục vụ cho việc xây dựng website và các GUI
Cực kỳ dễ sử dụng cho cả những người mới làm
quen với lập trình PHP đến những chuyên gia
trong lập trình PHP

Xây dựng ứng dụng web một cách nhanh chóng
và dễ dàng

Có một cộng đồng phát triển rộng lớn và nhiều
tài liệu hướng dẫn hỗ trợ



Yii Framework



Yii – Giới thiệu



Tác giả của Yii là Qiang Xue người Trung Quốc.

Qiang Xue bắt đầu xây dựng Yii vào khoảng
01/01/2008

Yii viết tắt của từ Yes, it is.

Is it fast? ... Is it secure? ... Is it professional?. Câu trả lời là “Yes it is!”.

Yii – Ưu điểm



Yii viết bởi HTML 5.0 rất rõ ràng và mạnh mẽ, phát triển tốt trên nền tảng Web 2.0, sử dụng tối đa các thành phần để tăng tốc độ viết ứng dụng.

Ưu điểm của Yii là tính đơn giản

Yii cũng được viết bởi những người viết “thực tế” hơn Zend nên có sẵn những công cụ “hợp thời trang” nhất cho người viết ứng dụng.



Yii ưu điểm



Yii là framework MVC nhưng có kiến trúc OOP rất tốt và định hướng theo component nên mức độ tái sử dụng giữa các project rất cao.

Yii chạy nhanh hơn Zend và Symfony.

Kích thước: 18.06 MB



Framework

CodeIgniter



Giới thiệu



- Codeigniter được đánh giá là một Framework mạnh mẽ với nhiều thư viện hỗ trợ người dùng từ dễ đến khó trong việc phát triển ứng dụng web.
- Để tiếp cận và triển khai CodeIniter chúng ta phải nắm vững những kiến thức liên quan đến lập trình hướng đối tượng và cách tổ chức vận hành code bằng mô hình MVC



Giới thiệu



- Mã nguồn của **CodeIgniter** nhỏ gọn chỉ 1.17MB (không tính phần User Guide). So với các PHP **framework** khác như **CakePHP** (1.3MB), **Yii** (18.06MB) hay **Zend Framework** (23.03MB)...kích thước của **CodeIgniter** giúp giảm thiểu đáng kể không gian lưu trữ.



Giới thiệu



- **CodeIgniter** được đánh giá là PHP **framework** có tốc độ nhanh nhất hiện nay
- Cấu trúc **URL** của CodeIgniter rất thân thiện với các máy tìm kiếm
- Cơ chế kiểm tra dữ liệu chặt chẽ, ngăn ngừa **XSS** và **SQL Injection**
- **CodeIgniter** hỗ trợ lập trình **AJAX**



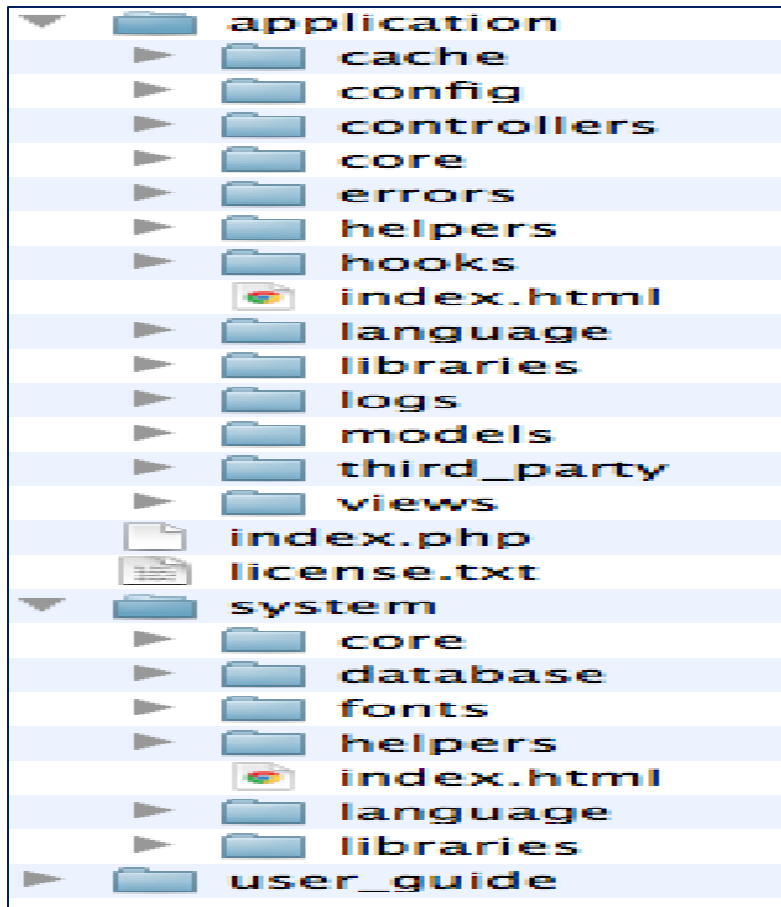
Cài đặt



- Tải CodeIgniter tại địa chỉ <http://codeigniter.com/download> phiên bản 4.0
- Giải nén, và chép vào thư mục **htdocs** của **Xampp**



Cấu trúc thư mục CodeIgniter



Danh sách file và thư mục



- ***index.php*** được xem như controller đầu vào
- **System**: thư mục chứa các thành phần cốt tạo nên CI, chứa driver, chứa tập tin cấu hình hệ quản trị CSDL, các thư viện có sẵn của CI.
- ***cache***: bộ đệm của hệ thống, chứa các trang đã được xử lý trước đó
- ***helper***: Chứa các hàm hỗ trợ cho lập trình viên khi viết ứng dụng.



Danh sách thư mục (tt)



- **libraries**: Chứa các thư viện dựng sẵn
- **Application**: thư mục dành cho người lập trình, tất cả các mã nguồn giao diện, xử lý của người lập trình
- Trong thư mục **Application** chứa các thư mục sau:



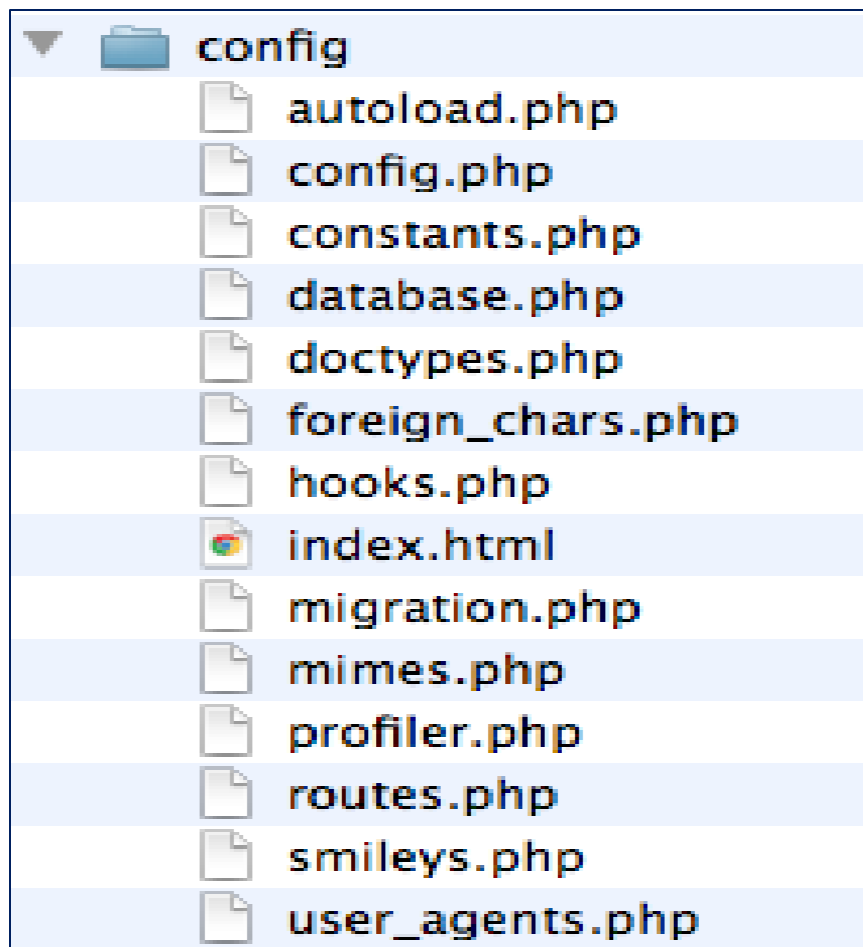
Các thư mục trong Application

- ***config***: Chứa các tập tin cấu hình hệ thống
- ***controllers***: chứa các lớp controller
- ***models***: chứa các lớp model .
- ***views***: chứa các lớp view.
- ***helpers***: chứa các hàm tiện ích do người dùng định nghĩa
- ***Language***: chứa các tập tin ngôn ngữ .
- ***libraries***: chứa các thư viện cho người dùng dùng định nghĩa

Cấu hình CodeIgniter

- Mở thư mục **config** trong thư mục **application** ta thấy được các tập tin cấu hình:

Các tập tin trong config



Cấu hình tập tin config.php



- **\$config['base_url']**: Xác định thư mục gốc chứa ứng dụng.
- **Ví dụ**: Sau khi ta chép thư mục CodeIgniter và htdocs và đổi tên thành "*quanlycasi*" thì lúc đó ta xác định base_url như sau:

```
$config['base_url'] = 'http://localhost/  
quanlycasi'
```



Cấu hình tập tin config.php (tt)

- Dùng hàm: **base_url()** để lấy đường dẫn gốc này
- **\$config['index_page']**: Xác định lại tập tin index, nếu đổi tên tập tin ***index.php*** thì ta dùng biến này để cấu hình lại cho phù hợp

Cấu hình tập tin **routes.php**



- Dùng để xác định **controller** bắt đầu thi hành khi chạy website
- `$route['default_controller'] = "tên controller"`
- CodeIgniter mặc định:

`$route['default_controller'] = "wellcome"`



URL trong CodeIgniter



- Một URL trong CodeIgniter có dạng:
base_url/index.php/controller/method/param/...
 - ❖ **Segment controller:** tên của lớp controller được gọi.
 - ❖ **Segment method:** tên của phương thức trong lớp controller ở trên.
 - ❖ **Segment param:** các đối số của phương thức đó



Ví dụ URL



*localhost/showbiz/index.php/casi/
exist_singer/1*

- URL trên có nghĩa:
 - ❖ phương thức gọi đến **exist_singer()** thuộc lớp **Casi** của **Controller**
 - ❖ Đối số truyền vào bằng **1**



MVC trong CodeIgniter



- Thư mục **controllers**: Chứa các tập tin thuộc thành phần controller của ứng dụng.
- Thư mục **models**: Chứa các tập tin thuộc thành phần Model của ứng dụng.
- Thư mục **views**: Chứa các tập tin thuộc thành phần Views trong ứng dụng



Định nghĩa lớp trong Model



```
class Tên_model extends CI_Model
{ //phương thức khởi tạo
    function __construct()
    {
        parrent::__construct();
    }
    function Function($biến)
    {
        .....
    }
}
```



Định nghĩa lớp trong Model(tt)

- Tên được viết hoa chữ đầu tiên, phần còn lại viết thường.
 - **Ví dụ:** `User_model`
- Tên tập tin được đặt như tên lớp, và được đặt trong thư mục `application/models/`
`application/models/user_model.php`
- Bắt buộc phải kế thừa từ lớp `CI_Model`. Trong hàm tạo của lớp con, phải gọi đến hàm tạo của lớp cha.

Gọi sử dụng Model



- Controller gọi sử dụng model theo cú pháp sau:
`$this->load->model('model_name');`
- **model_name**: tên của lớp **model**



Ví dụ



- Cách dùng “*model*” trong “*controller*”.
Ta tạo tập tin “*student.php*” trong thư mục “*application/models*”. Nội dung tập tin “*student.php*” như sau



Định nghĩa lớp Model



```
<?php
class Student extends CI_Model
{ public $ID;
  public $Name;
  public $Age;
  public function Set($id, $name, $age){
    $this->ID=$id;$this->Name= $name; $this->Age=$age;
  }
  public function PrintST(){
    echo "ID:". $this->ID."<br>";
    echo "Name:". $this->Name."<br>"; echo
    "Age:". $this->Age."<br>";
  }
}
?>
```



Cách Controller gọi Model



- Trong file “*usingstudent.php*” trong thư mục “application/controller

```
<?php
class Usingstudent extends
    CI_Controller{
    public function index(){
        $this->load->model("student");
        $s=new student();
        $s->Set("11250001","Mai Linh",19);
        $s->PrintST();
    }
}
?>
```



View trong CodeIgniter



- Dùng để hiển thị dữ liệu khi cần thiết.
- View được chỉ thị thi hành từ Controller theo cú pháp:
- `$this->load->view('view_name', $data);`
 - ❖ **view_name**: tên của view
 - ❖ `$data` chứa các dữ liệu sẽ được hiển thị trong **view**



Ví dụ gọi View trong Controller



- *"login.html"* được lưu trong thư mục "application/views" như sau:

```
<body>
  <form method="POST">
    Username: <input type="text"
      name="username"><br>
    Password: <input type="text"
      name="password"><br>
    <input type="submit" value="Login">
  </form>
</body>
```



Controller gọi thi hành View



- Trang *"login.php"* trong thư mục *"application/controller"* có nội dung như sau:

```
<?php
class Login extends
CI_Controller {
    public function index(){
        //sử dụng View "Login.html"
        $this->load->view('Login.html');
    }
}
?>
```


Kết quả



- Truy cập địa chỉ:
<http://localhost/CodeIgniter/index.php/login>

Username:

Password:

Login



Truyền dữ liệu qua View



Truyền dữ liệu thông qua mảng:

```
$data = array('name' => 'Linh',  
  'ID' => '11520001',  
  'age' => '33'  
);  
$this->load->view('student_detail', $data);
```

Khi ấy View lấy thông tin thông qua tên phần tử:

Ví dụ: `echo $name, $ID, $age`

Định nghĩa Controller



```
class Example extends CI_Controller
{    //phương thức tạo
    function __construct()
    {
        parent::__construct();
    }
    function method($param)
    {
        ...
    }
}
```

Nguyên tắc định nghĩa lớp controller



- Tên lớp được viết hoa chữ đầu tiên, phần còn lại viết thường. Ví dụ: **User**
- Tên tập tin được đặt giống tên lớp
- Đặt trong thư mục **application/controllers/**
- Phải thừa kế lớp **Controller** có sẵn
- **phương thức** có giới hạn truy xuất **private** trong **controller** được bắt đầu bằng ký tự gạch dưới “_”



Phương thức index trong lớp



- CodeIgniter dùng phương thức **index()** làm đầu vào của **controller**
- Phương thức này sẽ được gọi tự động trong trường hợp đối **segment** thứ hai của **URL** bị bỏ trống
- <http://localhost/index.php/user> của controller sau



Phương thức index trong lớp



```
class User extends CI_Controller
{
    function __construct()
    {
        parent::__construct();
    }
    function index()
    {
        echo 'Chào User';
    }
}
```



Library trong CodeIgniter



- CodeIgniter cung cấp bộ thư viện được xây dựng sẵn
- Các thư viện được chứa trong thư mục `system/libraries`.
- Bộ thư viện của CodeIgniter gồm:



Library



Thư viện	Công dụng
calendar	Hỗ trợ tạo lịch
cart	Hỗ trợ chức năng giỏ hàng
driver	Cho phép điều hướng
email	Hỗ trợ gửi email
encrypt	Hỗ trợ mã hóa và giải mã dữ liệu
form_validation	Kiểm tra dữ liệu đầu vào
ftp	Hỗ trợ FTP
image	Xử lý ảnh
javascript	Hỗ trợ javascript
log	Hỗ trợ ghi log
pagination	Phân trang tự động



Library



parser	Hỗ trợ Template Parser
profiler	Hiển thị kết quả Benchmark.
session	Hỗ trợ Session
sha1	Hỗ trợ Sha1
table	Hỗ trợ table
trackback	Cho phép nhận thông tin Trackback
typography	Định dạng văn bản
unit_test	Hỗ trợ unit testing
upload	Hỗ trợ upload tập tin lên Server
user_agent	Xác định thông tin trình duyệt, thiết bị di động, robot đang truy cập
xmlrpc	Cho phép gọi yêu cầu đến một xml-rpc hoặc tự xây dựng 1 xml-rpc cho hệ thống
zip	Hỗ trợ file zip



Library – cách dùng



- Dùng cú pháp:

```
$this->load->library('tên thư viện');
```

- Sử dụng các phương thức của thư viện sau khi đã load thư viện

```
$this->tên_thư_viện->method();
```



Ví dụ dùng library



- Sử dụng thư viện hỗ trợ tạo lịch "*calendar*" để hiển thị lịch của một tháng trong năm. Ta tạo tập tin "*showcalendar.php*" trong thư mục "*application/controllers*", trong tập tin ta định nghĩa lớp "*Showcalendar*" và phương thức "*index*". Tập tin có nội dung sau:



Ví dụ (tt)



```
<?php
    class Showcalendar extends CI_Controller {
public function __construct() {
    parent::__construct();
}
public function index() {
//load thư viện “calendar”
    $this->load->library('calendar');
//gọi sử dụng phương thức “generate” để hiện thị lịch tháng 10 năm 2014
    echo $this->calendar->generate(2014, 10);
}
}
?>
```



Kết quả



Ta truy cập địa chỉ: <http://localhost/giaotrinh/quanlygiaovien/index.php/showcalendar>



Tạo thư viện mới



- Thư viện do lập trình viên tạo mới sẽ được lưu trong thư mục application/libraries.
- Tên tập tin phải được viết hoa ký tự đầu tiên
- Ký tự đầu tiên của tên lớp phải viết hoa
- Tên lớp và tên tập tin chứa lớp phải trùng nhau

Sử dụng thư viên có sẵn trong thư viện mới



- Thư viện do lập trình viên tạo mới sẽ được lưu trong thư mục application/libraries.
- Tên tập tin phải được viết hoa ký tự đầu tiên
- Ký tự đầu tiên của tên lớp phải viết hoa
- Tên lớp và tên tập tin chứa lớp phải trùng nhau



Ví dụ:



- Ta tạo ra thư viện “*mylibrary*” bằng cách tạo một tập tin “*mylibrary.php*” trong thư mục “*application/libraries*”, trong tập tin chứa một lớp “*Mylibrary*” lớp “*Mylibrary*” chứa hàm “*showtable*”, hàm *showtable* sử dụng thư viện “*table*” của **CodeIgniter**, tập tin có nội dung như sau:

Sử dụng thư viện mới



- `$this->load->library("tên thư viện");`
- `echo $this->tên thư viện->method();`



Nội dung *mylibrary.php*



```
<?php
class Mylibrary extends CI_Controller{
    public function showtable()
    {
        $CI =&get_instance();
        $CI->load->library('table');
        $data = array(
            array('Name', 'Color', 'Size'),
            array('Fred', 'Blue', 'Small'),
            array('Mary', 'Red', 'Large'),
            array('John', 'Green', 'Medium')
        );
        echo $this->table->generate($data);
    }
}
```

?>

Dùng thư viện mới



- Dùng thư viện “*mylibrary*” do chúng ta tạo ra bằng cách tạo một tập tin “*usinglibrary.php*” trong thư mục “*application/controllers*”, nội dung tập tin như sau:



Nội dung file Usinglibrary.php



```
<?php  
    class Usinglibrary extends CI_Controller  
    public function index(){  
        $this->load->library('mylibrary');  
        echo $this->mylibrary->  
        >showtable();  
    }  
}  
?>
```



Kết quả



[http://localhost/giaotrich/quanlygiaovien/
index.php/usinglibrary](http://localhost/giaotrich/quanlygiaovien/index.php/usinglibrary)



CSDL trong CodeIgniter



- **Bước 1:** Cấu hình tập tin **database.php** trong thư mục **application/config**
- **Bước 2:** Kết nối CSDL
- **Bước 3:** Chuẩn bị và thực thi câu lệnh truy vấn



Cấu hình tập tin database.php



- **hostname:** tên host, mặc định là *"localhost"*
- **username:** Tên đăng nhập vào CSDL, mặc định là *"root"*.
- **password:** Mật khẩu đăng nhập vào CSDL, mặc định là *""*.
- **database:** tên CSDL, trong hình trên CSDL là *"quanlygiaovien"*
- **dbdriver:** trình quản trị CSDL: **MySQL**



Kết nối CSDL



- Ta dùng hàm: `database()` để kết nối CSDL theo cú pháp sau:

```
$this->load->database();
```



Chuẩn bị và thực thi lệnh truy vấn



- Cách 1:

```
$str="select * from giaovien";
```

```
$rs=$this->db->query($str)
```

- Cách 2:

Dùng các phương thức trong đối tượng db

- `$this->db->phương thức ();`

Câu lệnh chọn



- Lất tất cả các thuộc tính

- Cách 1:

- ```
$query = $this->db->get('tablename');
```

- Cách 2:

- ```
$this->db->from("mytable");
```

- ```
$query=$this->db->get();
```

- Lấy một số thuộc tính

- ```
$this->db->select('title, content, date');
```

- ```
$query = $this->db->get('mytable');
```



# Câu lệnh chọn



- Chọn có điều kiện

- ❖ `$this->db->where('name', $value);`  
`$query = $this->db->get('mytable');`
  - ❖ name: Tên thuộc tính
  - ❖ \$value: Giá trị điều kiện
- ❖ Đặt nhiều điều kiện ta dùng:
  - ❖ `$this->db->where('name', $name);`  
`$this->db->where('title', $title);`  
`$this->db->where('status', $status);`

# Câu lệnh chọn



- Chọn với phép kết (join)

```
$this->db->select('*');
```

```
$this->db->from('blogs');
```

```
$this->db->join('comments', 'comments.id = blogs.id');
```

```
$query = $this->db->get();
```



# Xử lý kết quả trả về của lệnh chọn

- Câu lệnh truy vấn:
  - ❖ `$str="select * from giaovien";`  
`$rs=$this->db->query($str);`
- Lấy kết quả trả về:
  - ❖ `foreach($rs->result() as $row)`  
`echo $row->tên field;`

# Lệnh chọn (tt)



- Câu lệnh truy vấn sử dụng
  - ❖ `$query = $this->db->get("giaovien");`
- Lấy kết quả trả về
  - `$object= query->result_object();`
  - `foreach($object as $value){`
  - `echo $value->tên field;`
  - `}`

# Group By



| id | user_id | points |
|----|---------|--------|
| 1  | 12      | 48     |
| 2  | 15      | 36     |
| 3  | 18      | 22     |
| 4  | 12      | 28     |
| 5  | 15      | 59     |
| 6  | 12      | 31     |

```
$this->db->select('user_id, COUNT(user_id) as
total');
```

```
$this->db->group_by('user_id');
```

```
$this->db->order_by('total', 'desc');
```

```
$this->db->get('tablename', 10);
```

| USER_ID | TOTAL |
|---------|-------|
| 12      | 3     |
| 15      | 2     |
| 18      | 1     |



# Having



- `$this->db->having('user_id = 45');`

- Truyền vào một mảng:

```
$this->db->having(array('title =' => 'My Title', 'id
<' => 45));
```





# Order by



- **Asc**: Sắp xếp tăng
- **Desc**: Sắp xếp giảm
  - ✧ `$this->db->order_by("title", "desc");`
  - ✧ `$this->db->order_by('title desc, name asc');`

# Distinct



- `$this->db->distinct();`
- `$this->db->get('table');`

# Lệnh xóa



```
$this->db->where('name', $value);
$this->db->delete("tablename");
```



# Lệnh thêm (insert)



- Cách 1: dữ liệu lấy từ mảng

```
$data = array(
 'title' => 'My
name title' ,
 'name' => 'My
Name' , 'date'
=> 'My date'
);
$this->db->insert('mytable', $data);
```

# Lệnh thêm (tt)



- Cách 2: dữ liệu lấy từ đối tượng

Giả sử ta định nghĩa một lớp như sau:

```
class Myclass {
 var $title = 'My Title';
 var $content = 'My Content';
 var $date = 'My Date';
}
```

✧Tiến hành thêm

```
$object = new Myclass();
$this->db->insert('mytable', $object);
```

# Cập nhật dữ liệu (update)



- Cách 1: Dữ liệu lấy từ mảng

```
$data = array(
 'title' =>
 $title,
 'name' =>
 $name,
 'date' =>
 $date
);
//điều kiện cập nhật dữ liệu
$this->db->where('id', $biển);
$this->db->update('mytable', $data);
```

# Cập nhật dữ liệu (tt)



- Cách 2: Dữ liệu lấy từ đối tượng

Giả sử ta có định nghĩa của một lớp như sau:

```
class Myclass {
 var $title = 'My Title';
 var $content = 'My Content';
 var $date = 'My Date';
}
```

```
$object = new Myclass();
```

*//Điều kiện cập nhật dữ liệu*

```
$this->db->where('id', $biến);
```

```
$this->db->update('mytable', $object);
```



# Xử lý kết quả trả về



- Câu lệnh truy vấn **INSERT**, **UPDATE**, **DELETE** trả về kết quả “true” nếu thành công và ngược lại trả về “false”





# Tìm kiếm theo điều kiện



```
$this->db->where($nameField,$value);
$query = $this->db->get('table');
if($query->num_rows() >= 1)
{
 return true;
}
else
 return false;
```



# Ví dụ Quản lý nhân viên



- Cấu hình:
  - ❖ Để sử dụng hàm "**base\_url()**" ta mở tập tin "*autoload.php*" trong thư mục "*quanlynhanvien/application/config*"
    - ❖ **\$autoload**['helper'] = **array('url');**



# Ví dụ Quản lý nhân viên



- Để khi truy cập địa chỉ <http://localhost/quanlynhanvien> ứng dụng sẽ thi hành phương thức "*index*" trong controller "*Home*". Ta cấu hình tập tin "*routes.php*" trong thư mục "*application/config*" như sau:
  - `$route['default_controller'] = "home";`
    - "*home*": Tên của controller

# Lập trình Ajax trong CodeIgniter

- CodeIgniter dùng 2 phương thức để gửi yêu cầu lên Server khi sử dụng kỹ thuật lập trình Ajax với JQuery
  - Phương thức GET
  - Phương thức POST



# Lấy dữ liệu trả về từ Serever



```
$.get("get.php?ten="+giá trị,function(data,status){
 $("#iddiv").html(data);
});
```

- **data**: Dữ liệu trả về từ Server
- **status**: status = “success” khi Server xử lý xong yêu cầu.
- **\$("#iddiv").html(data)**: Hiển thị dữ liệu trong tag <div>



# Dùng phương thức POST



- Dùng hàm post để gửi yêu cầu lên server:

`$.post(URL,data,callback);`

- URL: Tập tin xử lý yêu cầu
- Data: dữ liệu gửi lên server
- Callback: hàm nhận giữ liệu trả về



# Hàm nhận kết quả trả về



- Theo cú pháp sau:

```
function(data,status){
 $("#iddiv1").html(data);
};
```

- **data**: Dữ liệu trả về từ Server
- **status**: Trạng thái của yêu cầu (Request),  
status = "*success*" khi yêu cầu được xử lý xong.

# Ví dụ



Xét CSDL quản lý nhân viên gồm có các quan hệ sau:

- **CONGTY**(MaCongTy, TenCongTy, DiaChi)
  - **Tân từ:** Một công ty gồm có Mã công ty, tên công ty, địa chỉ. Mã công ty dùng để phân biệt giữa các công ty. Một công ty có nhiều chi nhánh
- **CHINHANH**(MaChiNhanh, TenChiNhanh, DiaChi, MaCongTy)
  - **Tân từ :** Một chi nhánh có mã chi nhánh (chuỗi), tên chi nhánh (chuỗi), địa chỉ (chuỗi). Mã chi nhánh để phân biệt giữa các chi nhánh. Một chi nhánh có nhiều phòng ban.



# Ví dụ



- **PHONGBAN**(MaPhong, TenPhong, MaChiNhanh)
  - **Tân từ:** Một phòng ban có mã phòng ban (chuỗi), tên phòng ban (chuỗi). Một phòng ban có 1 mã số để phân biệt với phòng ban khác. Một phòng ban có nhiều nhân viên.
- **NHANVIEN**(MaNhanVien, TenNhanVien, LuongThang, GioiTinh, MaPhong)
  - **Tân từ:** Một nhân viên có mã nhân viên (chuỗi), tên nhân viên (chuỗi), lương tháng (số thực), giới tính (true/false). Một nhân viên có 1 mã số để phân biệt với các nhân viên khác.

# Yêu cầu



- Dùng kỹ thuật lập trình Ajax trong CodeIgniter để liệt kê các chi nhánh của công ty. Tên công ty load từ CSDL, khi người dùng thay đổi mục chọn trong combobox thì Website liệt kê danh sách chi nhánh của công ty được chọn vào một bảng như hình sau:

# yêu cầu



## Danh sách công ty

CNTT1 ▼

### Danh sách chi nhánh

| Mã chi nhánh | Tên chi nhánh         |
|--------------|-----------------------|
| CN01         | Thành Phố Hồ Chí Minh |
| CN02         | Hà Nội                |
| CN03         | Đà Nẵng               |
| CN05         | Điện máy SG           |

# Cách giải



Bước 1: tạo model CongTy và lưu có nội dung sau:

```
class CongTy extends CI_Model{
 private $maCongTy;
 private $tenCongTy;
 private $diaChi;
 public function __construct() {
 parent::__construct();
 $this->load->database();
 $this->maCongTy="";
 $this->tenCongTy="";
 }
}
```

# Cách giải



- Phương thức trả về danh sách các đối tượng

```
public function getListObject($strFieldName = NULL,
$strWhere = NULL)
{
 if (!is_null($strWhere)) {
 $this->db->where($strWhere, NULL, FALSE);
 }
 if (!is_null($strFieldName)) {
 $this->db->select($strFieldName);
 }
 $query = $this->db->get("CONGTY");
 return $query->result_object();
}
```



# Cách giải



Bước 2: tạo model ChiNhanh và lưu có nội dung sau:

```
class ChiNhanh extends CI_Model{
 private $maChiNhanh;
 private $tenChiNhanh;
 private $diaChi;
 private $maCongTy;
 public function __construct() {
 parent::__construct();
 $this->load->database();
 $this->maChiNhanh="";
 $this->tenChiNhanh="";
 $this->maCongTy="";
 }
}
```

# Cách giải



- Phương thức trả về danh sách các đối tượng

```
public function getListObject_ajax($strFieldName
= NULL, $strWhere = NULL)
{
 if (!is_null($strWhere)) {
 $this->db->where("MaCongTy", $strWhere);
 }
 if (!is_null($strFieldName)) {
 $this->db->select($strFieldName);
 }
 $query = $this->db->get("CHINHANH");
 return $query->result_object();
}
```

# Cách giải



- Bước 2: thêm 1 phương thức CongTy\_Ajax trong controller "home"

```
public function CongTy_Ajax(){
 $congTy = new CongTy();
 $data['congty'] =
 $congTy->getListObject("*", "1=1");

 $this->load->view('viewcongty_ajax', $data);
}
```



# Cách giải



- Bước 3: tạo view có tên 'viewcongty\_ajax'

```
<h5 class="test">Danh sách công ty</h5>
<select id="macty">
<?php
foreach($congty as $value)
{ echo "<option value='$value-
>MaCongTy'>".$value->TenCongTy;
}
?>
</select>
<div id="iddiv">
</div>
```



# Cách giải



- thêm phần xử lý Ajax vào 'viewcongtv\_ajax'

```
$(".document").ready(function() {
 $("#macty").change(function(){
 var macongtv=$("#macty").val();
 $.post("<?php echo
base_url('index.php/home/LietKeChiNhanh_AJax');?>",
 {
 macongtv:macongtv
 },
 function(data,status){
 if(status=="success")
 {
 $("#iddiv").html(data);
 }
 });
 });
 });
});
```



# Cách giải



- Bước 4: Thêm phương thức **LietKeChiNhanh\_AJax** trong controller “home”

```
public function LietKeChiNhanh_AJax()
{
 $chinhanh = new ChiNhanh();
 $macongty=$_POST['macongty'];
 $data['chinhanh']=$chinhanh-
>getListObject_ajax("*",$macongty);

 $this->load-
>view('viewchinhanh_ajax.php',$data);
}
```



# Cách giải



- Bước 5: Thêm 1 view có tên **viewchinhanh\_ajax.php**

```
<table border="1">
 <tr>
 <th>Mã chi nhánh</th>
 <th>Tên chi nhánh</th>
 </tr>
 <?php
 foreach($chinhanh as $value)
 {
 echo "<tr>";
 echo "<td>".$value->MaChiNhanh."</td>";
 echo "<td>".$value->TenChiNhanh."</td>";
 echo "</tr>";
 }
 ?>
</table>
```



# Cách chạy ví dụ Ajax



[http://localhost/codeIgniter/index.php/home/congty\\_ajax](http://localhost/codeIgniter/index.php/home/congty_ajax)

# Giỏ hàng trong CodeIgniter



- CodeIgniter cung cấp thư viện “card” dùng để lưu giỏ hàng (shopping card) bằng session
- Cách dùng: Load thư viện shopping card theo cú pháp:
  - `$this->load->library("cart");`



# Các hàm trong thư viện card



- Thêm một sản phẩm vào giỏ hàng
  - `$this->cart->insert($data)`: hàm trả về true nếu thêm vào thành công, false nếu thêm vào không được
  - Ví dụ:

```
$data=array(
 "id" => "1",
 "name" => "Quạt điện",
 "qty" => "1",
 "price" => "100000",
 "option" => array("author" => "Samsung"),
); // Thêm sản phẩm vào giỏ hàng
if($this->cart->insert($data)){
 echo "Thêm thành công"; }
else{ echo "Thêm thất bại"; }
```



# Các hàm trong thư viện card



- Khi dùng hàm thêm thì tên các key trong mảng bắt buộc đặt tên: **id**, qty, **price**, name.
- Hiển thị danh sách các sản phẩm trong giỏ hàng:
  - `$this->cart->contents();`





# Các hàm trong thư viện card



- Xóa 1 sản phẩm có trong giỏ hàng, ta cần lấy được thuộc tính `rowId` của sản phẩm đó trong giỏ hàng, sau đó gán cho đối tượng số lượng sản phẩm là `qty = 0`
- `$this->cart->update(array('rowid' => $rowid, 'qty' => 0));`



# Các hàm trong thư viện card



- Xóa tất cả sản phẩm trong giỏ:
  - `$this->cart->destroy();`
- Tổng số tiền thanh toán trong giỏ hàng:
  - `$this->cart->total();`
- Lấy số lượng sản phẩm có trong giỏ hàng
  - `$this->cart->total_items();`



# Ví dụ minh họa



## Bước 1: tạo model có tên Shop\_model

```
<?php
class Shop_model extends CI_Model{
 public function getList(){
 return
 array(array('id' => '1', 'name'=>'Quạt điện','qty' =>'2','price' => '9000'),
 array('id' => '2', 'name' => 'Đèn học', 'qty' => '5', 'price' => '20000'),
 array('id' => '3', 'name' => 'Quần áo', 'qty' => '3', 'price' => '200000'),
 array('id' => '4', 'name' => 'Cặp sách', 'qty' => '1', 'price' => '500000'));
 }
}
?>
```

# Ví dụ minh họa (tt)



## Bước 2: Tạo controller có tên Shop

```
<?php
 defined('BASEPATH') OR exit('No direct script access
allowed');
 class Shop extends CI_Controller {
 public function __construct(){
 parent::__construct();
 }
 public function index() {
 $data['list'] = $this->cart->contents();
 $this->load->view('view_cart', $data);
 }
}
```



# Ví dụ minh họa (tt)



```
public function insert()
{
 $this->load->model('shop_model');
 $list = $this->shop_model->getList();
 if ($this->cart->insert($list))
 {
 echo "Thêm sản phẩm thành công";
 }
 else
 {
 echo "Thêm sản phẩm thất bại";
 }
}
```

# Ví dụ minh họa (tt)



```
public function delete($rowid)
{
 $this->cart->update(array('rowid' => $rowid, 'qty' => 0));
 $this->my_function-
>php_redirect(BASE_URL.'/index.php/shop/index');
}
public function deleteAll()
{
 $this->cart->destroy();
 echo "Done";
}
}
}
?>
```

# Ví dụ minh họa (tt)



Bước 3: tạo View có tên: `view_card.php`

```
<table border="1" width="700px" style="text-align: center; margin: 1px 300px;">
 <tr>
 <td>Index</td> <td>Name</td>
 <td>Quantity</td> <td>Price</td>
 <td>Amount</td> <td>Action</td>
 </tr>
```



# Ví dụ minh họa (tt)



```
<?php
$count = 1;
foreach ($list as $p)
{
 ?>

 <tr> <td><?php echo $count++; ?></td> <td>
 <?php echo $p['name']; ?>
 </td> <td><?php echo $p['qty']; ?></td>
 <td><?php echo number_format($p['price']); ?></td>
 <td><?php echo number_format($p['subtotal']); ?></td>
 <td><a href=" ../shop/delete/<?php echo
 $p['rowid'];?>">
 Delete</td> </tr>
<?php } ?>
```





# Ví dụ minh họa (tt)



```
</table>
<div style="margin-left: 300px;">
<h2>Tổng giá:
<?php
 echo $this->cart->format_number(
 $this->cart->total());
?>
</h2>
</div>
```

# kết quả



[http://localhost/shopping\\_card/index.php/shop](http://localhost/shopping_card/index.php/shop)

## Your Cart

Index	Name	Quantity	Price	Amount	Action
1	Quạt điện	2	10,000	20,000	<a href="#">Delete</a>
2	Đèn học	5	20,000	100,000	<a href="#">Delete</a>
3	Quần áo	3	200,000	600,000	<a href="#">Delete</a>
4	Cặp sách	1	500,000	500,000	<a href="#">Delete</a>

**Tổng giá: 1,220,000.00**



End!